Is it possible to ensure strong data guarantees in highly mobile networks?

Daniela Tulone
MIT CSAIL
tulone@csail.mit.edu

Abstract— **Ensuring the consistency and the availability of replicated data in highly mobile ad hoc networks is a challenging task because of the lack of a backbone infrastructure. Previous works provide strong data guarantees by limiting the motion and the speed of the mobile nodes during the entire system lifetime, and by relying on assumptions that are not realistic for most mobile applications.**

In this paper we provide a small set of mobility constraints necessary to ensure strong data guarantees. Our constraints can be applied also to low density mobile networks and to applications where the speed and the motion of the mobile nodes are unknown and they can change suddenly, such as vehicular networks. Our mobility model allows us to implement a read/write atomic shared memory that is able to guarantee data availability and atomic consistency despite high node mobility and node failures. Our implementation is provably correct and it can be applied for instance to energy management and to task coordination, as we show in the paper.

I. INTRODUCTION

Ensuring the availability and the consistency of shared data is a fundamental task for many ad hoc mobile network applications. For instance, nodes can share data containing configuration information, which is crucial for carrying out cooperative tasks. The shared data can be used for example to coordinate the duty cycle of mobile nodes to reduce energy consumption while maintaining network connectivity. The consistency and the availability of the data plays a crucial role in this case since the loss of information regarding the sleep/awake cycle of the nodes might compromise the network connectivity. The consistency and availability of the shared data is also relevant when tracking mobile objects, or in disaster relief applications where mobile nodes have to coordinate distributed tasks without the aid of a fixed communication infrastructure. This can be attained via read/write shared memory: each node maintains a copy of the shared memory (e.g., containing rescue information or data regarding the damage assessment), and dynamically updates it by issuing write operations. Also in this case it is important that the data produced by the mobile nodes does not get lost, and that each node is able to retrieve the most up–to–date information. Strong data consistency guarantees can be applied also to road safety, detection and avoidance of traffic accidents or safe driving assistance.

The atomic consistency guarantee introduced by Herlihy and Wing [9], is the most common data guarantee used in distributed systems because it ensures that the distributed operations (e.g., read and write operations) performed on the shared memory are ordered consistently with the natural order of their invocation and response time, and that each local copy reflects such an order. Intuitively, this implies that each node is always able to retrieve the most up-to-date copy. The implementation of a fault-tolerant atomic read/write shared memory represents a challenging task in *highly mobile* networks because of the lack of a fixed infrastructure, or nodes that can serve as a backbone. In fact, it is hard to ensure that each write request reaches a sufficiently large subset of nodes (in order to be retrieved), if nodes move over time according to unknown speed and paths.

The *focal point model* introduced by Dolev et al. in [4], provides a first answer to this challenge since it masks the dynamic nature of mobile ad hoc networks by a *static model*. More precisely, it associates abstract mobile nodes to fixed geographical locations called *focal points*. According to this model, a focal point is *active* at some point during the system lifetime if its geographical location contains at least one active mobile node. As a result, a focal point becomes *faulty* when each mobile node populating that region leaves it or crashes. The merit of this model is to study node mobility in terms of failures of *static* focal points, and to design coordination protocols for mobile networks in terms of static abstract nodes. The latter task is clearly easier than the former. However, the model proposed in [4] assumes that only a fraction of focal points can become faulty during the *entire* system lifetime. This implies that only a fraction of geographical subregions can become empty at some point during the system lifetime. Clearly, this assumption poses strong limitations on the motion of the mobile nodes over the system lifetime, and on the density of the network. Moreover, this condition is very difficult to ensure in mobile *sparse networks* where a node can trigger a focal point failure each time it leaves a focal point region to join another one. Note that the implementation of a read/write atomic shared memory proposed in [4] relies on these assumptions and therefore it is not resilient to high mobility.

In this paper we investigate a small set of mobility constraints that are necessary to ensure strong data guarantees. We employ the focal point model [4], which allows us to study node mobility in terms of focal point failures and to apply fault-tolerant techniques. Our goal is to devise mobility conditions that are sufficient to derive strong data guarantees and that are still realistic for applications involving high mobility, such as disaster relief applications or vehicular applications. In these applications nodes move according to unknown paths and speed. The key idea of our proposal consists of transforming the problem of tolerating high node mobility into the problem of tolerating *continuous* focal point failures, and applying fault–tolerance techniques, such as proactive recovery. In contrast with [4], our goal is to tolerate an unlimited number of focal point failures, that is to allow mobile nodes to move according unknown paths and speed. More precisely, our mobility model does not impose any limitation on node mobility during the system lifetime but during a time interval that is equal to the maximum round trip delay τ between any two nodes. Our mobility constraint depends on a parameter representing the *minimum coverage* of the nodes across the geographic system area. Note that this condition is weaker than assuming a specific node density for the system since some geographic subregions be more populated than others or they can be empty. As a result, our model allows higher node mobility with respect to previous work [4], and it is more realistic.

As mentioned before, our mobility model allows us to implement a read/write atomic shared memory that is resilient to high node mobility and node failures. It is built on top of the focal point model and tolerates unbounded focal point failures over the system lifetime. A key idea to guarantee the robustness of our implementation (data availability and atomic consistency) is represented by the *recovery* of the focal point after a failure. Our recovery protocol allows a previously faulty focal point to become active by retrieving the most up–to–date copy of the shared memory. Note that since the motion of the mobile nodes is continuous over time, nodes can leave a geographical subregion and join another one thus causing continual failures of the focal points. Therefore, it is crucial for the availability of the data that each focal point successfully recovers their state, and that at any time in the execution there is a *sufficient number* of *active* focal points. In fact, if the failure rate of the focal points exceeds the recovery rate at some point in the execution, the system can fall into a *stale condition* where the number of active focal points is not sufficient for the recovery to complete. In this case the data becomes unavailable. As a result, the availability of the data is strictly related to the *liveness* of the recovery protocol, and to its *response time*. The fact that our mobility constraints are easier to guarantee in a real setting, contributes to enhancing the robustness of our implementation and make it more practical.

Our contributions can be summarized as follows:

- We propose a small set of mobility constraints necessary to ensure strong data guarantees in highly mobile networks. These conditions are easier than previous work to guarantee in real mobile applications since they do not limit the motion of the nodes during the entire system lifetime.
- We propose an implementation of a read/write atomic shared memory based on our model, that is resilient to high node mobility and node failures. Our implementation guarantees data availability and atomic consistency, and it is provably correct. It is built on top of a novel implementation of the focal point, which is probabilistic and reduces the amount of communication and collisions relative to a focal point region.
- We briefly show how our implementation can be applied to energy management.

Structure of the paper. In Section II we compare our proposal with previous work, and in Section III we describe our system model, which consists of two abstractions: the *node layer* abstraction and the *application layer* abstraction. Section IV illustrates briefly our implementation of the focal point. In Section V we illustrate our mobility model, and in Section VI we briefly describe our implementation of a read/write atomic shared memory. In Section VII we apply our results to energy management, and then conclude by discussing future extensions.

II. RELATED WORKS

As mentioned in the Introduction, our work uses the focal point model proposed by Dolev et al. [4] which associates abstract mobile nodes to fixed geographical locations. However, they consider a weak mobility model that imposes strong limitations on the node motion and density over the entire system lifetime. This model is not realistic for most mobile applications and for low density networks. Our implementation relaxes these assumptions, and assumes *arbitrary* node motion. Note that our work does not rely on reconfigurations to guarantee fault–tolerance in highly mobile settings as in [6]. This leads to a simpler and more efficient implementation, features that are important in sensor networks due to their limited energy source.

Several solutions have been proposed for data dissemination in mobile ad hoc networks [8], [16], [19], [20]. However, their perspective is different than ours since they do not provide strong data consistency guarantees, such as atomic consistency and data availability. On the other side, some of these works [18], [20] addresses the problem of network partitions which we do not consider at this stage. Some of these proposals use node mobility to deliver information opportunistically: mobile nodes can exchange information when they meet [22], or move to deliver messages [18], thus improving the network connectivity.

III. SYSTEM MODEL

In this section we describe our system model which consists of two abstractions: a *node layer* abstraction consisting of mobile nodes, and an *application layer* abstraction built on top of the node layer. These abstractions share some similarities with [4].

A. Node layer abstraction

Our model consists of a bounded region $\mathbf{G}$ of a two-dimensional plane, populated by a dynamic set of mobile nodes. The mobile nodes can move on any continuous path in $\mathbf{G}$, and may fail at any time due to battery depletion or physical damage. They communicate with their neighbors through radio broadcast medium. We assume that each mobile node has a clock that is synchronized, and that each node is aware of its current position. This can be ensured by equipping the node with a GPS, or by applying location services such as [23].

Let us denote the physical broadcast radius of the nodes by r. We assume a reliable broadcast service, called the LBcast service, built on top of the physical radio broadcast. The broadcast radius of the LBcast service is equal to r. Note that by doing that we assume *symmetric* and *reliable* radio links. Although both assumptions are not entirely realistic, recent publications [25], [24] have proposed solutions for providing reliable broadcast in case of node mobility, and have shown that careful neighborhood management and retransmissions can provide loss rates as low as 1-2 percent in sensor networks, which should be sufficient for our purposes. We denote by d the maximum transmission delay of the LBcast service. Therefore, after d time units each neighbor receives any broadcast message.

We consider a set of geographical subregions of $\mathbf{G}$, called *focal point regions*, populated by mobile nodes. A mobile node is in a focal point region at a certain time if its position falls in that region. As mentioned above, our focal point model diverges from [4] for its underlying geometry. For instance, while the focal point regions proposed in [4] are defined as *non–intersecting regions* of $\mathbf{G}$, our focal point regions intersect. As shown in Section V-A, this assumption makes our implementation resilient to high node mobility and low density. We define the *diameter* of a closed geographic region $A \subset \mathbf{R}^2$ of the plane, as the maximum Euclidean distance between any two points in A, and denote by $C(P, c)$ the disk whose center is point P and radius $c \in \mathbf{R}$. The *proximity region* $Prox(A, \nu)$ of width ν of a closed region A with $\nu \in \mathbf{R}$, is defined as follows:

$$Prox(A,\nu)=\Big\{P : [P \in G \setminus A] \wedge [C(P,\nu) \cap A \neq \emptyset] \Big\}$$

It consists of points in $\mathbf{G} \setminus A$ whose distance from the border of A does not exceed ν. The proximity region is important to justify our failure model in case of high node mobility and low density (see Section V-A). We define now the focal point region and the n–region vector.

Definition 1: A focal point region *of* $\mathbf{G}$ *and* $\nu \geq 0$, *is a closed geographic region contained in* $\mathbf{G}$ *and whose diameter does not exceed* $r - 2\nu$. *A* n–region vector $\langle G_1, \ldots, G_n \rangle$ *for* $\mathbf{G}$ *and* ν *is a vector of* n*–focal–point regions of* $\mathbf{G}$ *such that* $\mathbf{G} = \bigcup_{i=1}^{n} G_i$.

Figure 1 graphically illustrates an example of focal point regions, more specifically a 28-region vector. The geographic system region $\mathbf{G}$ is partitioned into a square grid of focal point regions. The proximity region of the focal point region P is the surrounding section indicated in Figure 1.

Note that there is a strict connection between geometric properties and underlying mobile nodes. For instance, since the diameter of a focal point region does

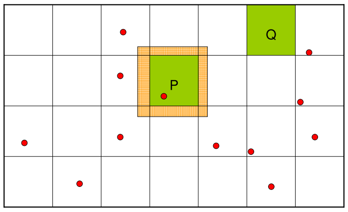

Fig. 1. Focal points.

not exceed r, all mobile nodes in a focal point region can communicate each other. Definition 1 implies that each mobile node is always contained in *at least one* of the focal point regions $G_1, \ldots, G_n$. Clearly, for a given choice of $\mathbf{G}, n$ and ν there might exist more than one n–region vector depending on the geometry of the focal point regions (e.g., disk, square).

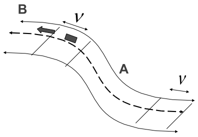

Fig. 2. Example of focal point.

An example: we can apply our model to vehicular networks. In this case the system region $\mathbf{G}$ is the union of the roads of a given geographical area. The focal point region is represented by road segments. Figure 2 shows two focal point regions A and B and their proximity region of width ν, and a car traveling in the proximity regions of A and B.

B. Application layer

The application layer is built on top of the node layer and consists of the following abstractions:

(1) A set of *stationary* focal points $F_1, \ldots F_n$ such that each F_i associates its focal point region G_i to the mobile nodes contained in G_i. Each focal point can be in one of the following modes: `inactive` if there are no correct nodes in G_i, `recov` if a node has joined its empty region G_i, and `active` otherwise. We say that F_i is *adjacent* to F_j if its associate focal point region G_i is adjacent to G_j. For instance, in Figure 1 P is an active focal point and Q is a faulty one.

(2) A set of mobile client nodes $C_1,\ C_2, \ldots, C_j, \ldots$ each associated with *at least* one focal point at any time in the execution. Each client can be in one of the following modes: `inactive`, `active` and `recov`. At any time only a finite number of clients can be in mode `active` or `recov`. This implies that mobile nodes can be replaced when they run out of battery.

(3) A virtual communication service VLBcast for F_i and C_j, built on top of the LBcast service and parameterized by radius $R > r$ and by the maximum round-trip delay τ. It allows them to communicate to a focal point using the adjacency of the focal point regions. Similarly to the LBcast service, the VLBcast service guarantees reliable delivery. It satisfies the following connectivity property: a focal point F_i (or client) is *connected* to F_j via VLBcast during $[t, t+\delta]$ with $\delta > 0$, if there exists a path of client nodes $C_0, \ldots, C_j, \ldots C_k$ during that interval, such that C_{i+1} is within the radio broadcast of C_i for any $1 \leq i < k$, and such that C_0 is contained in G_i and C_k in G_j.

In [4] a focal point F_i is *faulty* at time t if its location G_i contains no active client node at time t. However, this definition seems incomplete since it does not take into account the connectivity of the network. In fact, by active focal point we mean a focal point that is able to participate to the protocols. However, according to the previous definition F_i can be active but unable to communicate with other focal points (e.g., because its adjacent focal points they are all faulty). For this reason, we use the following definition:

Definition 2: A stationary focal point F_i is faulty at time t if G_i does not contain any active client node or if it is not connected to M focal points where M is a network implementation-dependent parameter contained in $(1, n)$.

As a result, at any time F_i is in one of the following modes: (1) `faulty`, according to Definition 2, (2) `recov`, if at least one active client node enters an empty focal point region G_i, or its connection is recovered, and F_i is in the process of recovering its state, (3) `active`, otherwise.

IV. IMPLEMENTING FOCAL POINTS

In this section we briefly describe our implementation of focal points for a shared read/write memory. As mentioned in the Introduction, each mobile node maintains a copy of the shared memory. Our goal is to “collapse” all the mobile nodes contained in a focal point region G_i at time t into a *virtual static* node associated to G_i. Note that the local copies of the mobile nodes contained in G_i must be consistent in order to achieve that. This can be guaranteed by using the geometry of the focal point regions and the reliability of the LBcast service (see Section III).

The underlying mobile nodes contained in G_i can follow different strategies each time a read/write request reaches G_i via the VLBcast service. A naive approach requires that each node in G_i replies. This approach is energy-consuming since the number of broadcasts performed is equal to the number of nodes currently contained in G_i, and it is likely to cause message collisions. An alternative implementation that addresses both problems relies on a *leader* node that is elected locally among the nodes contained in G_i at that time. However, this strategy involves a noticeable overhead especially in case of node mobility.

Our approach is probabilistic. Upon receiving a message each node waits for a random delay c before performing a broadcast, where c is uniformly chosen at random in $[0, \Lambda]$, and Λ is a network parameter (e.g., much larger than the maximum expected number of nodes in a focal point region). More precisely, a mobile node transmits a message m at time c only if none of its neighbors has transmitted m yet. This simple approach reduces the number of transmissions and collisions without the overhead of maintaining a leader since a node transmits only if required. However, this is done at the cost of higher communication latency of the VLBcast service.

The `join` protocol, run by each node upon entering a focal point region, is very important to determine the recovery of a focal point. A mobile node C triggers a recovery *as soon as* it passes the proximity region and enters into an *empty* focal point region (previous faulty region). More precisely, as soon as client C enters a new region G_i, it broadcasts a *join request* via LBcast, and waits for a reply. Since the LBcast service guarantees reliable delivery, if C does not receive any reply message within $\Lambda + d$ time units, where d is the time critical path for the LBcast service defined in Section III, it triggers a focal point recovery.

V. OUR MOBILITY MODEL

As mentioned in the Introduction we transform the problem of ensuring data consistency in case of high node mobility into the problem of ensuring data consistency in case of continuous static node failures (failures of the stationary focal points). Therefore, we tolerate high node mobility and unknown node motion and speed by tolerating a continuous and an unbounded number of focal point failures. Clearly, in order to derive strong data guarantees we need to define conditions regarding the failure rate of the focal points. In the following section we describe our failure model, and then discuss node mobility in the case of low density networks.

A. *Focal point failure model*

We tolerate an unbounded number of focal point failures by limiting the number of failures that can occur during a small time interval. Note that limiting the number of faulty focal points at any time is not sufficient to guarantee data availability since a focal point recovery completes only if a sufficient number of active focal points remains available during the time interval elapsed between its invocation and response time. Therefore, in order to guarantee data availability we need also to bound the failure rate between the invocation and response time of any distributed operation. In fact, if the failure rate exceeds the recovery rate at some point, the system can fall into a *stale* condition where focal points cannot complete their recovery. This case violates the availability of the data.

Our failure model is parametric in the maximum number f of failures that are tolerated by the system. This parameter depends on the specific implementation and it varies in $[0, n-3)$. We refer the reader to [17] for further details.

- A_1: at any time, there are at most f faulty focal points.
- A_2: at most α focal point failures can occur during τ time units, where $\alpha \geq 1$.

Although these assumptions look very similar, they are different in nature. In fact, assumption A_1 regards a *snapshot* of the system taken at a specific point in time. It provides an upper bound f on the number of faulty focal points present in the system at any time. This assumption is related to the geographical coverage of the mobile nodes in G since it assumes the existence of $n-f$ active and recovering focal points. Note that this condition does not imply that nodes are uniformly distributed in the remaining $n - f$ regions since some subregions can be highly populated and others could contain only one mobile node. Assumption A_1 says that the mobile nodes *cover* at any time at least $\frac{n-f}{n}$ fraction of the system region G and that $n - f$ subregions are connected. This assumption seems realistic for mobile applications where a majority of nodes move approximately according to some pattern or are task-driven (e.g., disaster relief applications, or monitor of animals such as herd). Note that at this stage of the work we do not consider network partitions.

Assumption A_2 provides an upper bound on the failure rate during τ time units, which is the maximum round-trip delay between any two nodes. This assumption is related to the density of the mobile nodes and their maximum speed. In fact, the probability that an active focal point F_i fails during τ time units is equal to the probability that each node contained in G_i crashes, or leaves the region during those τ time units. Therefore, if the speed of the mobile nodes is bounded by $[0, \frac{a}{\tau}]$, then the probability that F_i fails during τ time units is smaller than the probability that G_i contains only one node, which is at distance smaller than a from the border of G_i.

B. Sparse networks

Assumption A_2 is reasonable in most mobile networks, but it could be invalidated in case of sparse networks. (e.g., a mobile node travels *across the border* of some focal point region, thus causing frequent failures.) This occurs if during τ time units the only mobile node contained in G_i crosses more than α times the border of some focal point region leaving more than α empty subregions. This problem can be solved using the *proximity regions* defined in Section III. In fact, if the maximum speed of the node is $\frac{a}{\tau}$, then we can consider a set of proximity regions with $\nu = a$.

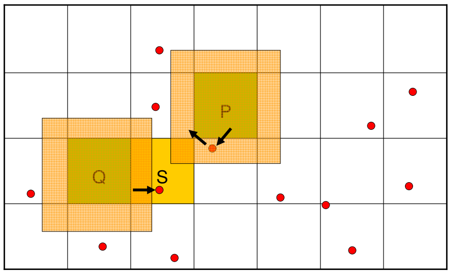

Fig. 3. Mobile nodes in the proximity regions.

According to our model each mobile node in a proximity region can communicate with each node in G_i and in any adjacent region. Therefore, if a node C contained in G_i leaves G_i and joins its empty adjacent region G_j, then C does not trigger a failure of F_i and a recovery of F_j if it is in $Prox(G_i, a)$. Figure 3 graphically illustrates the motion of two nodes during τ time units. The node motion is indicated by black arrows. The node previously contained in the focal point region of P enters first into the proximity region of P and then changes direction. Note that in both cases the node does not trigger any focal point recovery. The other node is initially contained in the proximity region of Q (it has not triggered yet a recovery of S). It triggers a recovery of S only upon leaving the proximity region of Q. In Figure 2 of Section III, focal point A remains active until the car leaves its proximity region.

VI. IMPLEMENTING A READ/WRITE ATOMIC MEMORY

In this section we briefly sketch our implementation of a read/write atomic shared memory. The read/write protocols are similar to [4]. Each mobile node maintains a copy of the state s associated to shared variable x, which is a compound object containing the value $s.val$ of x, the timestamp $s.t$ representing the time at which a client issued update $s.val$, and a `confirmed` tag that indicates if $s.val$ was propagated correctly. Each node can issue `write`, `read` and `recovery` operations. A new state is generated each time a client issues a write operation. A client requesting a write v computes a new state s consisting of value v and the current timestamp, and sends the new state s to each focal point (or to a subset). Upon receiving a write request, each non–faulty focal point (including recovering) replaces its state with the new state s only if the timestamp of s is higher than the timestamp of its local copy. A client issuing a read operation, requests a copy of focal point state from the other focal points (or from a subset), and computes the state with highest timestamp. As a result, the state remains unchanged during read or recovery operations. A recovery operation is treated as a read operation.

We have shown in [17] that our implementation satisfies data availability and atomic consistency: each mobile node is always able to retrieve the most up–to–date copy of the shared variable (the copy reflecting the last completed update). The main difficulty of our proposal consists of proving the data availability and the atomic consistency since these two properties strictly depend on each others. We broke this tie by having each node reply to a recovery request with its local state, and by proving first data availability. Due to space limitation we refer the reader to [17] for further details and for proofs.

Theorem 1: Our implementation of atomic read/write shared memory guarantees data availability and satisfies atomic consistency.

Note that since ensuring atomic consistency across the entire network can be energy-consuming, we can apply our implementation only to subregions of interest.

VII. AN APPLICATION: ENERGY MANAGEMENT

In this section we briefly sketch an application of our implementation of a read/write atomic shared memory that can be used to conserve energy in sensor networks.

Let us consider a sensor network where sensor nodes move over time within a geographic region G (e.g., sensors are placed on moving objects or animals, such as in ZebraNet [26]). We consider a n–region vector $\langle G_1, \ldots, G_n \rangle$ of G. Each sensor node maintains a n–vector V such that $V[i]$ contains information associated to the focal point region G_i. More precisely, $V[i]$ is a record consisting of two fields: $V[i].e$ represents the sum of the energy budget of the nodes containing in G_i at that time, and $V[i].r$ is a real number representing the compound reliability of the nodes containing in G_i (e.g., it is equal to 1 minus the probability that each node containing in G_i will fail in the next Γ time units, that is $1 - \Pi_{m \in G_i}\text{P(m fails)}$). Note that $V[i].e$ represents the energy availability associated with the focal point P_i and $V[i].r$ its reliability. Clearly, the information contained in $V[i]$ changes over time since nodes enter end leave focal point regions. Our implementation of a read/write atomic memory guarantees the correctness of the V vector. More precisely, each time a mobile node leaves a focal point region G_j and joins an adjacent region G_i it performs a `write/read` operation. It broadcasts a `join` message along with its energy budget. Each node contained in G_i and G_j update entries $V[i]$ and $V[j]$. Then, the head of P_i performs a `write` operation to propagate the updated $V[i]$ and $V[j]$ to the other focal points.

Note that the information contained in vector V can be applied to a number of critical network tasks. It can be used to coordinate the *low duty cycle* of mobile nodes and ensure network connectivity, or to improve the energy management of the network, thus routing messages through focal points that have higher energy budget. In addition, this strategy can be used to improve the robustness of communications when routing paths are chosen according to the resiliency of the focal points. Moreover, the information contained in vector V can be used to study the distribution of the mobile nodes (e.g., in ZebraNet the behavior of zebras).

VIII. CONCLUSIONS AND FUTURE EXTENSIONS

We have investigated a minimum set of mobility constraints that are necessary to ensure strong data guarantees in highly mobile networks. Our mobility model improves previous work for relaxing assumptions on the motion and speed of the nodes during the system lifetime. We proposed an implementation of atomic read/write memory based on our mobility model, which is provably correct.

There are several extensions that can be built on top of these results, as illustrated in [17]. For instance, our mobility model can be employed to design quorum systems [11] that are resilient to high node mobility. In [17] we provide a condition for quorum systems that is *necessary* to guarantee data availability and data consistency and that shows that quorum systems designed for static networks cannot guarantee strong data consistency guarantees in the presence of high node mobility. The design of quorum systems resilient to node mobility can clearly reduce the amount of transmissions and improve the load balancing, thus improving the energy consumption.

REFERENCES

[1] S. Bhattacharya. *Randomized location service in mobile ad hoc networks.* In Proc. of the 8th Intl. ACM Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pp. 66–73, Sep 2003.

[2] T. Camp, Y. Liu. *An adaptive mesh–based protocol for geocast routing.* J.Parallel and Distributed Computing: Special Issue on Mobile Ad–hoc Networking and Computing, pp. 196–213, 2002.

[3] S. Dolev, E. Schiller, and J. Welch. *Random walk for selfstabilizing group communication in adhoc networks.* In Proc. of the 21st Symp. on Reliable Distributed Systems, pp. 70–79, 2002.

[4] S. Dolev, S. Gilbert, N. Lynch, A. Shvartsman, J. Welch.*GeoQuorums: Implementing Atomic Memory in Mobile Ad Hoc Networks.* In Proc. of the 17th Intl. Conf. on Distributed Computing, pp. 306-320, October 2003.

[5] B. Englert and A. Shvartsman. *Graceful quorum reconfiguration in a robust emulation of shared memory.* In Proc. of the Intl. Conf. on Distributed Computer Systems, pages 454–463, 2000.

[6] S. Gilbert, N. Lynch, A. Shvartsman. *RAMBO II:: Rapidly reconfigurable atomic memory for dynamic networks.* In Proc. of the Intl. Conf. on Dependable Systems and Networks, pp. 259–269, June 2003.

[7] Z. J. Haas and B. Liang. *Ad hoc mobile management with uniform quorum systems.* IEEE/ACM Transactions on Networking, 7(2):228–240, April 1999.

[8] H. Hayashi, T. Hara, S. Nishio. *Updated data dissemination methods for updating old replicas in ad hoc networks.* In Personal and Ubiquitous Computing, 9(5), pp. 273-283, Sept 2005.

[9] M. Herlihy, J. Wing. *Linearizability: A correctness condition for concurrent objects.* ACM Trans. on Programming Languages and Systems, 12(3):463–492, July 1990.

[10] B. Liu, P. Brass, O. Dousse, P. Nain, D. Towsley. *Mobility improves coverage of sensor networks.* In Proc. of MobiHoc, pp. 300-308, May 2005.

[11] D. Malkhi, M. Reiter, A. Wool. *The Load and Availability of Byzantine Quorum Systems.* SIAM J. Comput. 29(6), pp. 1889-1906 (2000).

[12] R. Prakash, Z. Haas, M. Singhal *Load-balanced location management for cellular mobile systems using quorums and dynamic hashing.* Wireless Networks, 7(5), pp. 497-512, Sept 2001.

[13] G. Karumanchi, S. Muralidharan, R. Prakash. *Information dissemination in partitionable mobile ad hoc networks.* In Proc. of Symp. on Reliable Distributed Systems, pp. 4–13, 1999.

[14] Y. B. Ko, N. Vaidya. *Geotora: a protocol for geocasting in mobile ad hoc networks.* In Proc. of the Intl. Conf. on Network Protocols, pages 240–249, November 2000.

[15] J. C. Navas and T. Imielinski. *Geocast – geographic addressing and routing.* In Intl. Conf. on Mobile Computing and Networking, pp. 66–76, 1997.

[16] J. Luo, J-P. Hubaux, P. Eugster. *Resource management: PAN: providing reliable storage in mobile ad hoc networks with probabilistic quorum systems.* In Proc. of the 4th Intl. Symp. on Mobile ad hoc networking and computing, pp. 1-12, 2003.

[17] D. Tulone. *Mechanisms for energy conservation in wireless sensor networks.* Ph.D. thesis, Departement of Computer Science, University of Pisa, Dec 2005.

[18] W. Zhao, M. Ammar, E. Zegura. *A message ferrying approach for data delivery in sparse mobile ad hoc networks.* In Proc, of the 5th Intl. Sym. on Mobile ad hoc Networking and Computing, pp. 187-198, May 2004.

[19] K. Chen, S. Shah, K. Nahrstedt. *Cross-Layer Design for Data Accessibility in Mobile Ad Hoc Networks.* In Wireless Personal Communications, 21(1), pp. 49-76, Apr 2002.

[20] H. Wu, R. Fujimoto, R. Guensler, M. Hunter. *MDDV: a mobility-centric data dissemination algorithm for vehicular networks.* In Proc. of the 1st Intl. Workshop on Vehicular ad hoc Networks, pp. 47-56, Oct 2004.

[21] H. Kim, T. Abdelzaher, W. Kwon. *Minimum–energy asynchronous dissemination to mobile sinks in wireless sensor networks.* In Proc. of SenSys, pp. 193-204, Nov 2003.

[22] A. Vahdat, D. Beker. *Epidemic routing for partially–connected ad hoc networks.* Tech report, Duke University, 2000.

[23] A. Smith, H. Balakrishnan, M. Goraczko, N. Priyantha. *Support for location: Tracking moving devices with the cricket location system.* In Proc. of the 2nd Intl. Conf. on Mobile systems, applications, and services, Jun 2004.

[24] J. Polastre, J. Hill, and D. Culler. *Versatile low power media access for wireless sensor networks.* In Proc. of SenSys, 2004.

[25] M. Mohsin, D. Cavin, Y. Sasson, R. Prakash, A. Schiper. *Reliable Broadcast in Wireless Mobile Ad Hoc Networks.* In Proc. of the 39th Annual Hawaii Intl. Conf., vol. 9, issue , Jan 2006.

[26] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, D. Rubenstein. *Energy–efficient computing for wildlife tracking: design trade–offs and early experience with Zebranet.* In Proc. 10th Conf. on Architectural Support for Programming Languages and Operating Systems, Oct 2002.